Chapter 6

On Polyglot Programming in the Web

Juhana Harmanen
Ada Drive Ltd., Finland

Tommi Mikkonen
Tampere University of Technology, Finland

ABSTRACT

Different programming languages have been designed to solve problems efficiently in different domains. The goal of polyglot programming, a technique where several languages are used in the creation of a single application, is to combine and utilize the best solutions from different programming languages and paradigms in a seamless fashion. In this paper, the authors examine polyglot programming in the context of web applications, where it has been commonly used to create compelling applications, but where there is still considerable potential to improve development in various ways.

INTRODUCTION

Different programming languages have been designed to solve problems efficiently in different domains. For instance, SQL is commonly associated with databases, scripting languages are often used as glue that integrates different programs, and C is used for low-level code. When various programming languages are used in the creation of a single program, the term polyglot programming is used. The goal of polyglot programming is to combine the best solutions from different languages and paradigms in a seamless fashion.

A polyglot system has two essential aspects; the platform used for the integration and the programming languages that are supported by the platform. For instance, when using Java, the Java Virtual Machine (JVM) enables using other languages that build on the same runtime environment. Such languages include Scala and Groovy, which offer different programming paradigms than plain Java.

In this chapter, we examine polyglot programming in the context of web applications. While the approach is in extensive use in today’s web development, for example, embedded HTML and SQL or JavaScript with Cascading Style Sheets (CSS) and HTML, only limited research has been conducted on this topic.

DOI: 10.4018/978-1-4666-9916-8.ch006
On Polyglot Programming in the Web

BACKGROUND

The forced separation of languages in writing can be considered somewhat unnatural in this multilingual world. Authors of novels, for instance, sometimes break this separation because it is too restrictive, and journalists and historians combine philosophic prose with statistical facts (Beardsmore, 1978). The Russian author Leo Tolstoy, in his novel War and Peace written in Russian language, frequently borrowed words and phrases from French and German to be as expressive and effective in language as possible. Rare and inventive authors like J.R.R. Tolkien have even made up their own languages.

To some extent, programming languages share properties with natural languages. Each programming language has its own nuances that make it distinct from others. This can be seen as a justification for mixing languages to create a combination of characteristics needed for solving a particular problem that no single language can address elegantly.

Polyglot Programming

The term polyglot programming was first introduced in software development context in 2002 as a hypothesis to use several programming languages within one environment. Later authors tend to use slightly different approaches, for example, “programming in more than one language within the same context”, which delegates the definition to what the context is.

From the development perspective, the context is partly constituted by the number of teams and the fashion the resulting software is integrated. Polyglot programming takes place even if one of the teams uses a different programming language than others, and the integration between the parts developed using different languages is tight. However, when the teams need no information on languages other teams are using, polyglot programming no longer takes place. Instead, individual components constitute distinct entities – consider, for example, a service-to-service application in which only well-defined interfaces are needed.

Associated advantages. The definition of productivity is a much-debated aspect of programming languages. Common metrics include lines of code (LOC) and function points per unit time (Delorey et al., 2007; Maxwell et al., 1996; Sebesta, 2009). While it has been claimed that productivity does not depend on programming language (Brooks, 1995), Delorey et al. (2007) presents contradicting view – due to the nature of the problems within the scope of the productivity measurement, any findings from case studies are hard to generalize. Problems include human factors like motivation, skill and experience, and environmental factors like integrated development environment (IDE) and library support, and also factors related to geographical distribution of projects (Cataldo et al., 2008; Cataldo et al. 2007; De Souza, 2005; Sullivan et al. 2001).

Polyglot programming aims to combine and integrate the best solutions from different languages. Therefore, simpler solutions to the problems at hand become available (Fjeldberg, 2008). A suitable language for solving a particular problem will result in shorter solution in terms of LOC because of the built-in primitives and idioms. Following the assumption that developers produce the same amount of code lines regardless of language, polyglot programming allows them solve more problems – and be more productive (Brooks, 1995). In addition to reduced LOC, the thought process of a developer can be simplified as solutions emerge naturally in the appropriate languages. For example, taking advantage of static polymorphic type-checking in functional programming, a large class of errors on race conditions and deadlocks in communication between processes can be caught at compile time (Pareto, 2000;